Boundary Data Augmentation for Offline Reinforcement Learning

SHEN Jiahao[1,2], JIANG Ke[1,2], TAN Xiaoyang[1,2]
(1. College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China;
2. MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing 211106, China)

DOI: 10.12142/ZTECOM.202303005
https://link.cnki.net/urlid/34.1294.TN.20230814.1519.002.html, published online August 14, 2023
Manuscript received: 2023-07-05

Abstract: Offline reinforcement learning (ORL) aims to learn a rational agent purely from behavior data without any online interaction. One of the major challenges encountered in ORL is the problem of distribution shift, i.e., the mismatch between the knowledge of the learned policy and the reality of the underlying environment. Recent works usually handle this in a too pessimistic manner to avoid out-of-distribution (OOD) queries as much as possible, but this can influence the robustness of the agents at unseen states. In this paper, we propose a simple but effective method to address this issue. The key idea of our method is to enhance the robustness of the new policy learned offline by weakening its confidence in highly uncertain regions, and we propose to find those regions by simulating them with modified Generative Adversarial Nets (GAN) such that the generated data not only follow the same distribution with the old experience but are very difficult to deal with by themselves, with regard to the behavior policy or some other reference policy. We then use this information to regularize the ORL algorithm to penalize the overconfidence behavior in these regions. Extensive experiments on several publicly available offline RL benchmarks demonstrate the feasibility and effectiveness of the proposed method.

Keywords: offline reinforcement learning; out-of-distribution state; robustness; uncertainty

Citation (Format 1): SHEN J H, JIANG K, TAN X Y. Boundary data augmentation for offline reinforcement learning [J]. ZTE Communications, 2023, 21(3): 29 - 36. DOI: 10.12142/ZTECOM.202303005
Citation (Format 2): J. H. Shen, K. Jiang, and X. Y. Tan, "Boundary data augmentation for offline reinforcement learning," *ZTE Communications*, vol. 21, no. 3, pp. 29 - 36, Sept. 2023. doi: 10.12142/ZTECOM.202303005.

1 Introduction

Reinforcement learning (RL) is one of the major branches of machine learning that has been successfully applied in various fields in recent years, such as power grid control[1], recommendation systems[2], and robotics[3]. RL training usually involves a large number of try-and-error interactions with underlying systems. However, such "interaction hungry" behavior could have a serious negative impact on many real-world applications, especially when the online data are either costly or dangerous to collect, e.g., in healthcare[4], autonomous driving[5], and so on. To address this problem, the idea of offline RL (ORL) is to learn a new policy only from data based on offline dataset without online interaction. Unfortunately, the direct employment of the common off-policy strategy often fails to achieve the same level of performance as in the online setting[6 - 7].

The extrapolation error from out-of-distribution (OOD) actions is generally thought of as the main reason responsible for the aforementioned performance degradation[6]. The OOD actions here mean the actions taken by a model or system that is outside the range of the examples it was trained on. Since the dataset used for ORL training is generated by a behavior policy different from the new policy, possibly working in a different environment as well, it is not always possible for the agent to generalize the knowledge learned from the data to unseen real online situations. This is not uncommon in practice and usually manifests as the overestimated Q-value of OOD actions and such error compounds, leading to potentially dangerous consequences. The problem is usually referred to as distribution shift. To address this, many recent works, such as Conservative Q-Learning (CQL) [8] and Implicit Q-Learning (IQL)[9], take a conservative strategy by trying to prevent the agent from taking overestimated OOD actions, with the idea that if most of the states visited are familiar to us, the chance of error and the subsequent error compounding phenomenon could be greatly reduced. However, taking OOD actions online is almost inevitable, and avoiding the queries on them may significantly reduce the robustness of the agent. Actually, not all OOD data is non-generalizable[10].

This work is partially supported by the National Key R&D program of China under Grant No. 2021ZD0113203 and National Science Foundation of China under Grant No. 61976115.

This observation motivates some works[11-12] to formulate the ORL problem as an uncertainty-penalizing policy optimization problem, where some self-supervised methods are usually adopted. These methods utilize the model prediction to provide the supervised signal for OOD data. However, such a prediction could be unreliable, highlighting the necessity of carefully balancing the tradeoff between the so-called radicalism and conservatism when dealing with OOD data.

In this paper, we propose a novel method, named Boundary Conservative Q Learning (Boundary-CQL; BCQL), to improve the robustness of the learned policy while keeping conservative when dealing with the OOD data. Our key idea is to weaken the agent's confidence in highly uncertain regions while doing offline learning. To find those regions, we propose simulating them with a modified Generative Adversarial Net (GAN) such that the generated data follow the same distribution as the old experience but are very difficult to be dealt with by themselves, with regard to some reference policies. In practice, the reference policy could be an empirical behavior policy or a pre-trained high-capacity policy, such as a CQL policy. As the found uncertain region generated data is visually located at the boundary of the distribution of the original data, we call them boundary OOD data. Finally, we learn the new policy via the Bellman operator while simultaneously maximizing its entropy at the generated boundary OOD data, hence decreasing the confidence in the estimation of the optimal actions. In this way, we effectively maintain a balance between the minimization of Bellman error and policy conservatism. Extensive experiments on several publicly available offline RL benchmarks demonstrate the feasibility and effectiveness of the proposed method.

It is worth noting that our method can also be thought of as a self-supervised offline RL method but is unsupervised in nature, as we only use the reference policy to identify the most difficult regions within the distribution defined by the training experience, without showing the algorithm the exact actions to be taken there. This is similar to those methods in machine learning that improve their generalization capability by negative sample mining[13-15] but has never been used in the case of offline RL, to the best of our knowledge.

In what follows, a brief review of related work is given in Section 2, and a concise introduction to the background of offline RL is provided in Section 3. The description of the boundary OOD data and BCQL method is presented in detail in Section 4. Experimental results are presented in Section 5 to evaluate the effectiveness and properties of the proposed method from multiple aspects. Finally, the paper concludes with a summary of the findings and contributions.

2 Related Work

In this section, we briefly review some works most relevant to our method in literature, which mainly involve offline RL methods and OOD simulation methods.

2.1 Offline Reinforcement Learning

Offline reinforcement learning is one of the hottest research directions in RL in recent years, where, as mentioned before, the major challenge is how to handle the distribution shift problem. This can be roughly divided into three categories: the policy constraint, uncertainty-based, and regularization of the value function. Regarding the policy constraint method, the BCQ algorithm[16] addresses the exploration error caused by the distribution shift through batch-constrained restriction. The bootstrapping error accumulation reduction (BEAR) algorithm[17] solves the problem of mismatch between the learning strategy, optimal strategy, and sampling strategy through a support set matching method and can achieve better results even when the sampling strategy is poor. The behavior regularized actor critic (BRAC) algorithm[18] tries to combine the advantages of both BCQ and BEAR algorithms for better learning efficiency. On the uncertainty-based approach, a typical representative method is the random ensemble mixture (REM)[19] method, which uses multiple parameterized Q functions to estimate Q values while enforcing Behrman consistency during learning.

CQL[8] is one of the most representative methods of the third category. It uses a regularization term to the traditional Q-value network so as to learn a relatively conservative Q function. However, since CQL sets a strict restriction on OOD action evaluation, it could be overly conservative. There are some works like IQL[9] and Mildly Conservative Q-learning (MCQ)[20] trying to fix this issue, where instead of directly learning actions out of data, known state action experience is used to learn how action values vary and future outcomes are averaged with random dynamics.

2.2 OOD Solution Based on Generative Model

OOD data in machine learning usually refer to the data that are significantly different from the training data on which a model is trained. These data are harmful to a machine learning model as they could mislead the model to make incorrect predictions. Hence, they have drawn the attention of many researchers in recent years[21]. Generative networks are useful tools for high-dimensional sampling and have been widely used to generate OOD data. The most popular generative models include adversarial generation networks[22] and the autoencoder/variational autoencoder[23].

The work based on auto-encoder/variational auto-encoder (PIDHORSKYI et al.)[24] used the variational auto-encoder to calculate reconstruction errors to identify OOD. First, the parametric manifold structure implied by the normal distribution is linearized to calculate OOD probability. Next, the method of probability decomposition is given, and then the local coordinates of the tangent space of the manifold are used to calculate the reconstruction error. Competitive Reconstruction Autoencoder (CoRA)[25] trained two autoencoders on the in-distribution and abnormal data, respectively, and used the

reconstruction error of the two autoencoders as the suspicious identification signal.

On the other hand, based on an adversarial generation network, Anomaly Detection with Generative Adversarial Networks (ADGAN)[26] generates OOD samples by checking whether the sampled data are satisfactory in a hidden space, while PNET[27] uses the generation network to generate and identify OOD samples based on their reconstruction errors.

2.3 Generative Model for Offline Reinforcement Learning

Generative models have been widely used in offline reinforcement learning for different usages. In BCQ[16] and BEAR[17] algorithms, conditional variational autoencoders generate data that satisfies the constraint. Action-conditioned Q-learning (AQL)[28] replaces the conditional variational autoencoder in BEAR with a residual generative model to improve fitting performance. MCQ[20] uses a generative model such as conditional GAN to estimate the sampling policy. Diffusion Q-learning[29] uses a diffusion model to constrain target policy and add a loss of maximum action value to the original diffusion loss. Selecting from Behavior Candidates (SfBC)[30] also uses a diffusion model combined with an in-sample planning technique to further avoid selecting out-of-sample actions and increase computational efficiency. Unlike prior work, we use a generative model to generate data not only following the same distribution as the experience but also in uncertain regions for agents.

3 Preliminaries

A Markov decision process (MDP) can be specified by a tuple $\mathcal{S}, \mathcal{A}, r, \mathcal{T}, \gamma$, where $\mathcal{S}$ regards the state space, $\mathcal{A}$ is the action space, $r: \mathcal{S} \times \mathcal{A} \to \mathcal{R}$ is the reward function, which is used to evaluate the action under state s, $\mathcal{T}: \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the transition, and γ represents the discount factor. Reinforcement learning aims to find a policy to maximize the expected cumulative rewards. Q function $Q^{\pi}(s,a) = \mathbb{E}_{\pi}[\sum_{t=0}^{\infty} \gamma^t r_t | s, a]$ measures the discounted long-term reward given the state-action pair (s,a) and the policy π. Q-learning is a classic method that trains the Q-value function by minimizing the Bellman error over Q[31]. In the setting of continuous action space, Q-learning methods use an exact or approximate maximization scheme, such as the cross entropy method (CEM)[32], to recover the greedy policy as follows:

$$Q \leftarrow \arg\min_{Q} \mathbb{E}[\hat{\mathcal{B}}^{\pi} Q(s,a) - Q(s,a)]^2,$$
$$\pi \leftarrow \arg\max_{\pi} \mathbb{E}_s \mathbb{E}_{a \sim \pi(\cdot|s)} Q(s,a), \quad (1)$$

where $\hat{\mathcal{B}}^{\pi} Q(s,a)$ represents the empirical Bellman target, defined as $\hat{\mathcal{B}}^{\pi} Q(s,a) = r(s,a) + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')} Q(s',a')$.

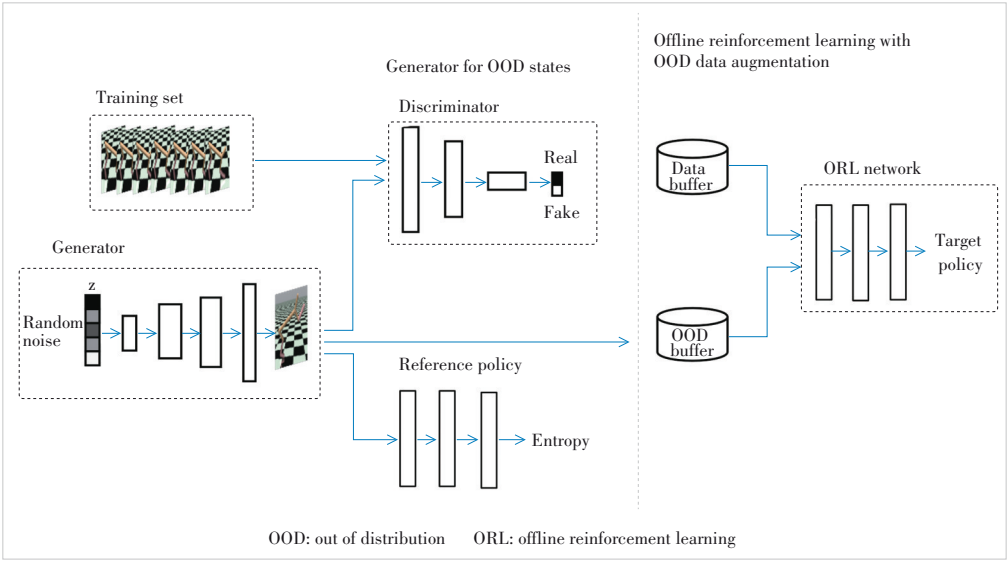
In the setting of offline reinforcement learning, Eq. (1) would be performed on a dataset $\mathcal{D}$, and the result is collected via a behavior policy π_{β}. Due to the aforementioned distribution shift issue, OOD queries usually yield incorrectly estimated Bellman targets. CQL, as a representative OOD-constraint offline RL algorithm, tries to underestimate the Q-values for OOD state-action pairs to prevent the agent from the extrapolation error[6] as follows:

$$Q \leftarrow \arg\min_{Q} \alpha \cdot \left(\mathbb{E}_{s \sim \mathcal{D}, a \sim \pi(a|s)}\left[Q(s,a)\right] - \mathbb{E}_{s,a \sim \mathcal{D}}\left[Q(s,a)\right]\right) + \frac{1}{2}\mathbb{E}_{s,a,s' \sim \mathcal{D}}\left[\left(Q(s,a) - \mathcal{B}^{\pi} Q(s,a)\right)^2\right], \quad (2)$$

where π is the new policy to be learned and α is the balance-coefficient. CQL tries to prevent the Q-value of OOD actions from being overestimated, but avoiding OOD queries would degrade the robustness when the agent faces an unseen state.

4 Method

In this section, we first introduce the boundary OOD data in our work. Then we illustrate the primary approach that is utilized to generate the boundary OOD data. Finally, we give the detailed design of the proposed method BCQL. The framework of our proposed method is shown in Fig. 1, where the



▲ **Figure 1. Overall architecture of the proposed Boundary Conservative Q Learning (Boundary-CQL, BCQL) method, where the left column illustrates the pipeline to generate boundary OOD data based on an adversarial generative model, while the right column performs Offline RL with the generated data**

generator is optimized via the discriminator and the pre-trained reference policy simultaneously, and then the generated data is provided for the offline RL algorithm to enhance its robustness.

4.1 Motivation

As mentioned before, to enhance the robustness of the new policy, it is necessary to generate more OOD data. Considering that not all the OOD data are generalizable in the offline setting, we generate certain OOD states that are not very far away from the training dataset and require that the pre-trained reference policy has high entropy in making decisions at these states. In other words, our generated data should have two features: 1) OOD, which means that the distribution should be different from the training dataset, such that the reference agent would be confused about what to do at these states, and 2) boundary, which means that the generated data should not be totally unrelated to the offline dataset. Hence we name this kind of OOD data as boundary OOD data.

4.2 Generating Boundary OOD Data

In this section, we describe how to generate boundary OOD data within the adversarial generative framework.

First, recall that the loss function of GAN is defined as follows:

$$\min_G \max_D \mathbb{E}_{P_{\text{in}}(x)}\left[\log D(x)\right] + \mathbb{E}_{P_{\text{pri}(z)}}\left[\log\left(1 - D\left(G(z)\right)\right)\right], \quad (3)$$

where G denotes the generator, D the discriminator, P_{in} refers to the distribution of in-distribution (ID) data while $P_{\text{pri}(z)}$ refers to some prior distributions such as a Gaussian distribution. The loss function minimizes the error rate of the discriminator for the real data while maximizing the error rate of the discriminator for the generated data until a Nash equilibrium is achieved.

To generate the desired boundary OOD data described in the previous section, we adopt the following loss function:

$$\min_G \max_D \left[\mathbb{E}_{P_{\text{in}}(s)}\left[\log D(s)\right] + \mathbb{E}_{P_{G_B}(s')}\left[\log\left(1 - D(s')\right)\right]\right] + \beta_G \mathbb{E}_{P_{G_B}(s')} H\left[\pi_{pre}(\cdot\,|s')\right], \quad (4)$$

where s' refers to the generated OOD state and π_{pre} refers to the pre-trained reference policy. The first term of Eq. (4) requires that the generated distribution $P_{G_B}(\cdot)$ should still follow the distribution $P_{\text{in}}(\cdot)$ of the offline dataset, but under the constraint defined by the second term of Eq. (4), which forces the generated data to satisfy the requirement that the reference policy has high entropy $H\left[\pi_{\text{pre}}(\cdot|s')\right]$ over them, where the entropy of the reference policy π_{pre} is defined as $H\left[\pi_{\text{pre}}(\cdot|s')\right] = \sum_a \pi_{\text{pre}}(a|s')\log \pi_{\text{pre}}(a|s')$. The tradeoff between the above two terms is controlled by another parameter β_G, which should be set based on the specific applications.

In implementation, we use Conditional GAN[33] to generate s' conditioned on the origin state s and use the pre-trained CQL as the reference policy π_{pre}.

4.3 Boundary Conservative Q-Learning

To enhance the robustness of the new policy learned offline, we aim to weaken its confidence at the generated boundary OOD states mentioned before. A direct way to realize this is to add regularization $\mathcal{L}_{\text{BCQL}}$ to maximize the entropy of the new policy when making decisions at these states:

$$L_{\text{BCQL}} = H\left[\pi\left(a\middle|\hat{s}\right)\right] = -\sum_a \pi\left(a\middle|\hat{s}\right)\log \pi\left(a\middle|\hat{s}\right), \quad (5)$$

where $\hat{s}$ is the boundary OOD state generated and π is the new policy. Then the whole loss function of the policy network is as follows:

$$\mathcal{L}_\pi = -\mathbb{E}_{s\sim\mathcal{D}, a\sim\pi(a|s)} Q(s,a) - \lambda \mathcal{L}_{\text{BCQL}}, \quad (6)$$

where λ is the balance-coefficient of the BCQL term, π is the new policy, and $\hat{Q}$ is the target Q network.

As Eq. (6) shows, we can keep the new policy conservative as commonly done in normal offline RL training, but it has to be uncertain as the reference policy does when facing the generated boundary OOD data. Then the loss function of Q networks can be formulated as follows:

$$\mathcal{L}_Q = \alpha \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{a\sim\pi(a|s)}\left[Q(s,a)\right] - \mathbb{E}_{a\sim\pi_\beta(a|s)}\left[Q(s,a)\right]\right] + \frac{1}{2}\mathbb{E}_{s,a,s'\sim\mathcal{D}}\left[\left(Q - \mathcal{B}^\pi Q\right)^2\right], \quad (7)$$

where α is the weight of the CQL term and π_β denotes the empirical behavior policy.

Algorithm 1. Boundary-CQL

Input: Q networks Q, pre-trained reference policy π_{pre}, prior distribution $p_{\text{pri}}(z)$, generator G_B, offline dataset $\mathcal{D}$, generative coefficient β_G, and OOD punishment coefficient λ

1: Train G_B via Eq. (4) using π_{pre} and β_G
2: **for** each iteration **do**
3: Sample a mini-batch $B = (s, a, r, s')$ from $\mathcal{D}$
4: Sample z from $p_{\text{pri}}(z)$ and generate $\hat{s} = G_B(z)$
5: Update the new policy π with $\hat{s}$ and B via Eq. (6)
6: Update the Q networks Q with B via Eq. (7)
7: **end for**
8: Output the new policy π

Algorithm 1 summarizes the main pipeline of the proposed

method. To be specific, we first train the generator G_B via Eq. (4), and then we enter the offline RL loop, where we update the policy network π via Eq. (6) and the Q networks Q via Eq. (7) alternately. Finally, we output the policy network π for the testing stage.

5 Experiments

This section starts with an introduction to the datasets used in our research. Subsequently, the efficacy of our proposed framework is demonstrated through its assessment on Datasets for Deep Data-Driven Reinforcement Learning (D4RL) benchmarks. Further, an examination of the behavior of our generator and its impact on the new policy is conducted. Finally, a sensitive analysis is performed to elucidate the contribution of each parameter.

5.1 Datasets

The experiments presented in this study are carried out on the OpenAIGYM subset of the D4RL[34] tasks. For performance evaluation, we utilize datasets that are a combination of multiple policies, namely medium, medium-replay, medium-expert, and expert. To ensure the robustness of our findings, we conduct all experiments at four distinct random seeds.

5.2 Comparative Study

To demonstrate the superiority of our proposed framework, we conduct a comparative analysis with several state-of-the-art algorithms, including behavior cloning (BC), BEAR[17], Soft Actor-Critic (SAC)[35], twin-delayed deep deterministic policy gradient (TD3)+BC[36], and CQL[8]. In particular, we obtain the results for CQL and BEAR using our implementation, while the effects of BC and SAC are taken from Clean Offline Reinforcement Learning (CORL)[37] and MCQ[20], respectively. Additionally, we obtain the results for TD3+BC from its original publication. Table 1 presents the results, with the highest mean value being denoted in bold.

As is evident from the results presented in Table 1, our proposed algorithm outperforms the other state-of-the-art approaches in the medium, medium-expert, and medium-replay datasets, which exhibit diverse characteristics. In comparison with the basic version of CQL, our approach demonstrates superior performance in estimating OOD data. However, in an expert setting, although we employ a generative network to simulate OOD data, with the constraint on the OOD data, our approach performs better than the original CQL but worse than TD3+BC when employing behavior cloning in the half cheetah task. Overall, these findings highlight the excellent efficiency of our proposed framework in both complex tasks and expert environments.

Fig. 2 displays the performance of BCQL and CQL during the training process. As illustrated by the curves, our proposed algorithm outperforms both BC and basic CQL in the medium, medium-replay, and medium-expert environments, owing to the utilization of augmented data. In contrast, BC employs data obtained through behavior cloning. In the expert environment, the regularization imposed on the generated low-confidence data leads to a minimal impact on the performance of high-confidence data.

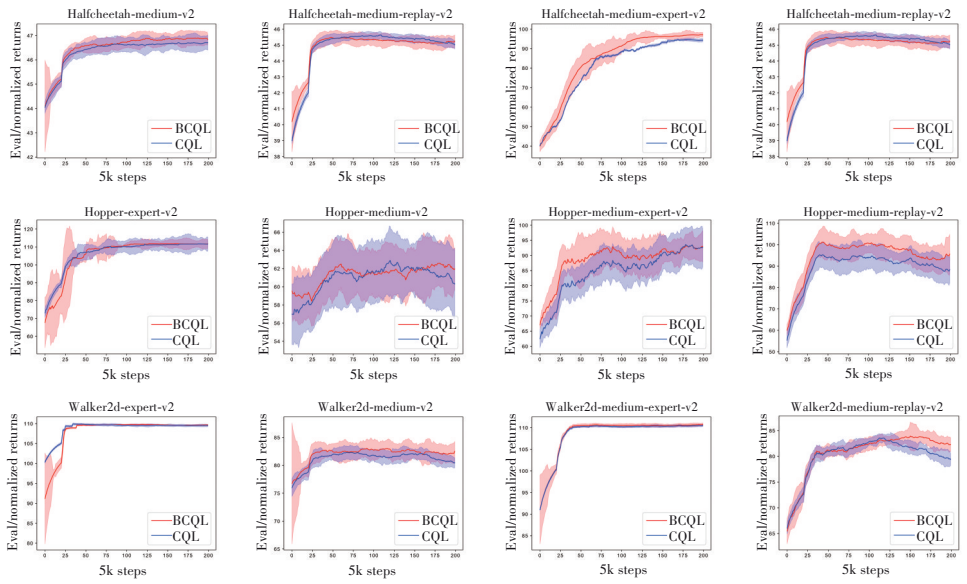
5.3 Behaviors of Generator

The present study employs a GAN-based generator to generate data in various environments, and the real and generated states are visualized, as shown in Fig. 3. The generated data is observed to be irregular yet maintained its validity in comparison to the original data. To assess the effectiveness

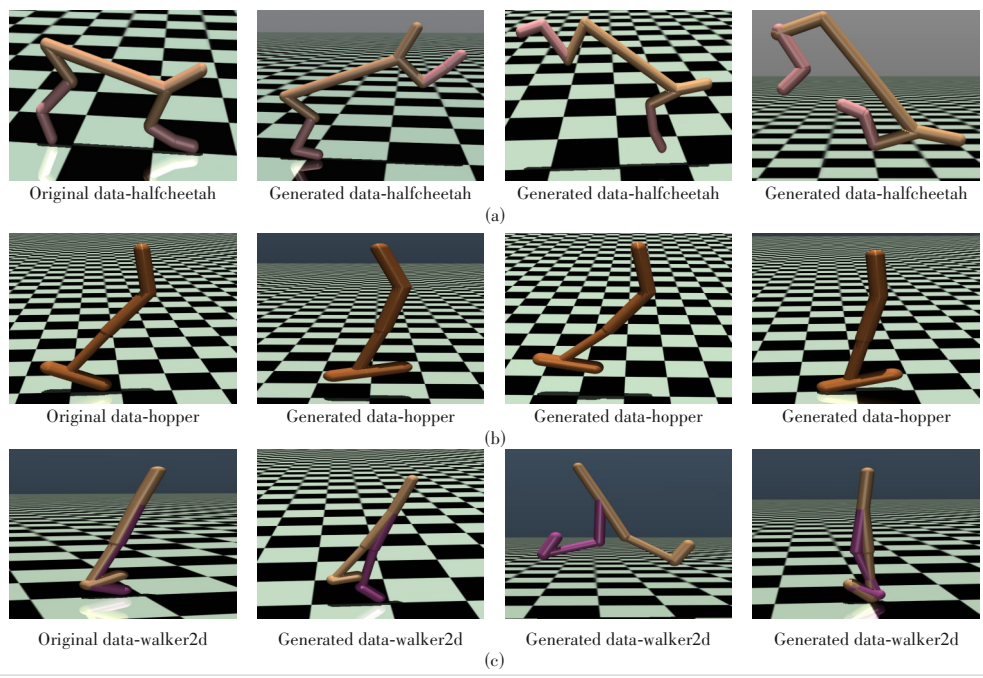
▼Table 1. Performance of BCQL and prior methods on MuJoCo tasks from D4RL, on the normalized return metric (the highest means are bolded)

Task Name	BC	BEAR	SAC	TD3+BC	CQL	BCQL
Halfcheetah-medium-v2	42.4±0.2	37.1±2.3	**55.2±27.8**	48.3±0.3	47.1±0.2	47.1±0.7
Hopper-medium-v2	53.5±2.0	30.8±0.9	0.8±0.0	59.3±4.2	64.9±4.1	**66.1±5.2**
Walker2d-medium-v2	63.2±18.8	56±8.5	-0.3±0.2	83.7±2.1	80.4±3.5	**84.6±2.5**
Halfcheetah-medium-replay-v2	35.7±2.7	36.2±5.6	0.8±1.0	44.6±0.5	45.2±0.6	**46.1±1.5**
Hopper-medium-replay-v2	29.8±2.4	31.1±7.2	7.4±0.5	60.9±18.8	87.7±14.4	**93.9±11.8**
Walker2d-medium-replay-v2	21.8±11.7	13.6±2.1	-0.4±0.3	81.8±5.5	79.3±4.9	**82.5±7.3**
Halfcheetah-medium-expert-v2	56.0±8.5	44.2±13.8	28.4±19.4	90.7±4.3	96±0.8	**97.5±3.2**
Hopper-medium-expert-v2	52.3±4.6	67.3±32.5	0.7±0.0	**98.0±9.4**	93.9±14.3	95.9±13.3
Walker2d-medium-expert-v2	99.0±18.5	43.8±6.0	1.9±3.9	110.1±0.5	109.7±0.5	**110.2±1.0**
Halfcheetah-expert-v2	91.8±1.5	**100.2±1.8**	-0.8±1.8	96.7±1.1	96.3±1.3	98.4±3.2
Hopper-expert-v2	107.7±0.7	108.3±3.5	0.7±0.0	107.8±7	109.5±14.3	**111.7±8.3**
Walker2d-expert-v2	106.7±0.2	106.1±6.0	0.7±0.3	**110.2±0.3**	108.5±0.5	109.7±1.0
Total average	63.3	56.2	7.9	82.7	83.8	87.0

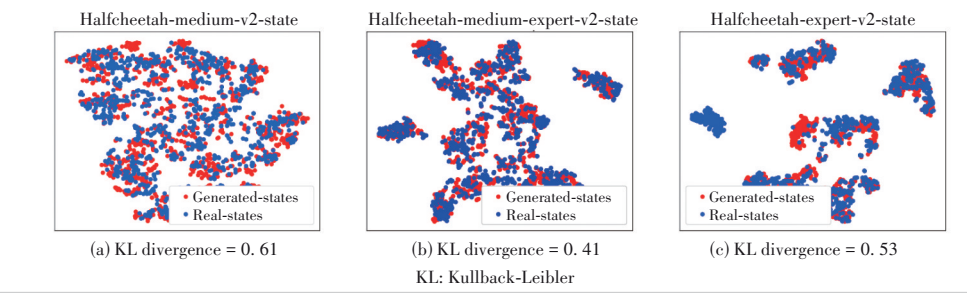
BC: behavior cloning
BCQL: Boundary Conservative Q Learning
BEAR: bootstrapping error accumulation reduction
CQL: Conservative Q-Learning
D4RL: Datasets for Deep Data-Driven Reinforcement Learning
SAC: Soft Actor-Critic
TD3: twin-delayed deep deterministic policy gradient



▲Figure 2. Policy performance during training in different environments



▲Figure 3. Visualization of data generated in different environments



▲Figure 4. Distribution of generated states and real states in different environments

of the generator, experiments are conducted in a halfcheetah environment. Fig. 4 demonstrates that the generated data closely approximates the original data, although the two are not identical. This finding satisfies the boundary requirements of the experiment. Moreover, the distribution of action possibility depicted in the figure indicates that the generated data adheres to the low confidence criteria of a pre-trained RL network. Thus, the generator is capable of producing data with the features described in Section 3.

5.4 Parameter Sensitivity Analysis

We conduct several experiments to evaluate the sensitivity of the following two parameters in our algorithm: the parameter β_G in Eq. (4) and the BCQL weight λ in Eq. (6).

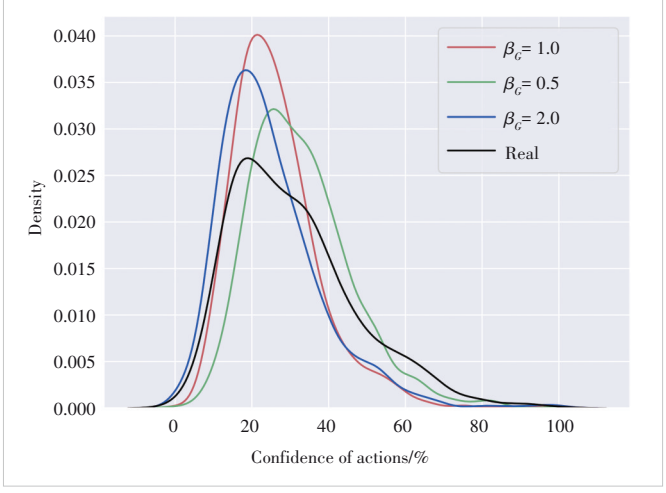
5.4.1 Study of Parameter β_G

The parameter β_G plays a crucial role in training the generator, as it affects its performance. We illustrate this by considering the task of walker2 d-medium. We set the iteration number K for the generator at a fixed value of 5 000 to avoid any confounding effects. As depicted in Fig. 5 and Table 2, the generator's performance is sensitive to changes in β_G. Specifically, increasing β_G leads to a decrease in the trained policy's confidence, while simultaneously increasing the KL divergence. The KL divergence represents the distance between the generated distribution and the original distribution. In order to strike a balance between being close to the original data and having low confidence, we use the smallest possible value for β_G that still

satisfies the low confidence requirement.

5.4.2 Study of Parameter λ

The parameter λ in Eq. (6) affects the behavior of the network when using generated data. Specifically, a higher value of λ results in a more conservative behavior, while a lower value leads to greater flexibility. To investigate the impact of λ on the performance of the network, we conduct experiments while keeping the other parameters constant, and the results are presented in Table 3, which indicates that a large value of λ can be detrimental to performance when the environment is diverse, and therefore, a milder value of λ may be more appropriate. Based on the results, a value of $\lambda = 1.0$ should be suitable in most situations.



▲**Figure 5. Confidence of actions on generated data under different β_G in a walker2d-medium environment**

▼**Table 2. KL divergence between generated states and origin states under different β_G in a walker2d-medium environment**

β_G	KL Divergence	β_G	KL Divergence
0.2	0.08	1.0	0.41
0.4	0.21	1.5	0.57
0.5	0.34	2.0	0.76

KL: Kullback-Leibler

▼**Table 3. Performance of BCQL with different λ, on the normalized return metric (the highest means are bolded)**

Task Name	λ = 0 (CQL)	λ = 0.5	λ = 1	λ = 1.5
Halfcheetah-medium-v2	**47.1±0.2**	46.8±1.3	47.1±0.7	46.1±2.2
Hopper-medium-v2	64.9±4.1	64.8±7.6	64.3±4.2	**66.1±7.2**
Walker2d-medium-v2	80.4±3.5	**84.6±2.5**	81.8±2.1	79.3±4.5
Halfcheetah-medium-replay-v2	45.2±0.6	45.0±1.0	44.9±0.5	**46.1±1.5**
Hopper-medium-replay-v2	87.7±14.4	90.2±10.5	**93.9±11.8**	83.5±14.4
Walker2d-medium-replay-v2	79.3±4.9	**82.5±7.3**	80.7±5.5	77.7±8.9
Halfcheetah-medium-expert-v2	96±0.8	95.2±0.4	**97.5±3.2**	96.9±1.1
Hopper-medium-expert-v2	93.9±14.3	94.0±8.7	**95.9±13.3**	94.8±11.3
Walker2d-medium-expert-v2	109.7±0.5	**110.2±1.0**	110.1±0.5	109.3±0.3
Halfcheetah-expert-v2	96.3±1.3	**98.4±3.2**	97.4±2.3	98.2±1.3
Hopper-expert-v2	106.5±14.3	109.7±7.9	**111.7±8.3**	111.2±10.2
Walker2d-expert-v2	108.5±0.5	108.7±0.3	**109.7±1.0**	109.5±0.5

BCQL: Boundary Conservative Q Learning CQL: Conservative Q-Learning

6 Conclusions

The proposed method BCQL improves the robustness of offline reinforcement learning algorithms while maintaining consistency with the original data distribution, based on a novel OOD simulation technique using a GAN. Extensive experiments are performed on several publicly available offline RL benchmarks, showing that the proposed BCQL method achieves state-of-the-art performance while maintaining high robustness and conservation. Our work highlights the benefits of improving the robustness of offline reinforcement learning algorithms, which is an important research direction given the increasing interest in offline RL applications.

References

[1] CHEN D, CHEN K A, LI Z J, et al. PowerNet: multi-agent deep reinforcement learning for scalable powergrid control [J]. IEEE transactions on power systems, 2022, 37(2): 1007 – 1017. DOI: 10.1109/TPWRS.2021.3100898
[2] CHEN X C, YAO L N, MCAULEY J, et al. A survey of deep reinforcement learning in recommender systems: a systematic review and future directions [EB/OL]. (2021-09-08)[2023-04-12]. https://arxiv.org/abs/2109.03540
[3] OLIFF H, LIU Y, KUMAR M, et al. Reinforcement learning for facilitating human-robot-interaction in manufacturing [J]. Journal of manufacturing systems, 2020, 56: 326 – 340. DOI: 10.1016/j.jmsy.2020.06.018
[4] YU C, LIU J M, NEMATI S, et al. Reinforcement learning in healthcare: a survey [J]. ACM computing surveys, 2023, 55(1): 1 – 36. DOI: 10.1145/3477600
[5] GRIGORESCU S, TRASNEA B, COCIAS T, et al. A survey of deep learning techniques for autonomous driving [J]. Journal of field robotics, 2020, 37(3): 362 – 386. DOI: 10.1002/rob.21918
[6] FUJIMOTO S, MEGER D, PRECUP D. Off-policy deep reinforcement learning without exploration [C]//International Conference on Machine Learning. PMLR, 2019: 2052 – 2062. DOI: 10.48550/arXiv.1812.02900
[7] LEVINE S, KUMAR A, TUCKER G, et al. Offline reinforcement learning: tutorial, review and perspectives on open problems [EB/OL]. (2020-05-04)[2023-04-12]. https://arxiv.org/abs/2005.01643
[8] KUMAR A, ZHOU A, TUCKER G, et al. Conservative Q-learning for offline reinforcement learning [EB/OL]. (2020-06-08)[2022-11-08]. https://arxiv.org/abs/2006.04779
[9] COUPRIE C, FARABET C, NAJMAN L, et al. Indoor semantic segmentation using depth information [EB/OL]. (2013-01-16)[2023-04-02]. https://arxiv.org/abs/1301.3572
[10] MCCRACKEN M W. Robust out-of-sample inference [J]. Journal of econometrics, 2000, 99(2): 195 – 223. DOI: 10.1016/S0304-4076(00)00022-1
[11] YU T H, THOMAS G, YU L T, et al. MOPO: model-based offline policy optimization [EB/OL]. (2020-05-27)[2022-10-08]. https://arxiv.org/abs/2005.13239
[12] GUO K Y, SHAO Y F, GENG Y H. Model-based offline reinforcement learning with pessimism-modulated dynamics belief [EB/OL]. (2022-10-13)[2023-04-02]. https://arxiv.org/abs/2210.06692
[13] WU C Y, MANMATHA R, SMOLA A J, et al. Sampling matters in deep embedding learning [C]//IEEE International Conference on Computer Vision (ICCV). IEEE, 2017: 2859 – 2867. DOI: 10.1109/ICCV.2017.309
[14] SHRIVASTAVA A, GUPTA A, GIRSHICK R. Training region-based object detectors with online hard example mining [C]//IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2016: 761 – 769. DOI: 10.1109/CVPR.2016.89
[15] ROBINSON J, CHUANG C Y, SRA S, et al. Contrastive learning with hard negative samples [EB/OL]. (2020-10-09) [2023-03-23]. https://arxiv.org/abs/

2010.04592
[16] FUJIMOTO S, MEGER D, PRECUP D. Off-policy deep reinforcement learning without exploration [EB/OL]. (2018-12-7) [2022-10-2]. https://arxiv. org/abs/1812.02900
[17] KUMAR A, FU J, TUCKER G, et al. Stabilizing off-policy Q-learning via bootstrapping error reduction [C]//33rd International Conference on Neural Information Processing Systems. Curran Associates Inc., 2019: 11784 – 11794. DOI: 10.48550/arXiv.1906.00949
[18] WU Y F, TUCKER G, NACHUM O. Behavior regularized offline reinforcement learning [EB/OL]. (2019-11-26) [2022-09-11]. https://arxiv. org/abs/1911.11361
[19] AGARWAL R, SCHUURMANS D, NOROUZI M. An optimistic perspective on offline reinforcement learning [EB/OL]. (2019-07-10)[2022-10-11]. https://arxiv.org/abs/1907.04543
[20] LYU J F, MA X T, LI X, et al. Mildly conservative Q-learning for offline reinforcement learning [EB/OL]. (2022-07-09)[2023-04-12]. https://arxiv. org/abs/2206.04745
[21] YANG J K, ZHOU K Y, LI Y X, et al. Generalized out-of-distribution detection: a survey [EB/OL]. (2021-10-21) [2022-03-01]. https://arxiv. org/abs/2110.11334
[22] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative adversarial networks [J]. Communications of the ACM, 2020, 63(11): 139 – 144. DOI: 10.1145/3422622
[23] KINGMA D P and WELLING M. Auto-encoding variational bayes [EB/OL]. (2013-12-20)[2023-02-14]. https://arxiv.org/abs/1312.6114
[24] PIDHORSKYI S, ALMOHSEN R, ADJEROH D A, et al. Generative probabilistic novelty detection with adversarial autoencoders [EB/OL]. (2018-07-06) [2023-04-10]. https://arxiv.org/abs/1807.02588
[25] TIAN K, ZHOU S G, FAN J P, et al. Learning competitive and discriminative reconstructions for anomaly detection [C]//33th AAAI Conference on Artificial Intelligence. ACM, 2019: 5167 – 5174. DOI: 10.1609/aaai.v33i01.33015167
[26] DEECKE L, VANDERMEULEN R, RUFF L, et al. Image anomaly detection with generative adversarial networks [C]//European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases. ECMLPKDD, 2018. DOI: 10.1007/978-3-030-10925-7_1
[27] ZHOU K, XIAO Y T, YANG J L, et al. Encoding structure-texture relation with P-net for anomaly detection in retinal images [M]//Computer Vision – ECCV 2020. Cham, Switzerland: Springer international publishing, 2020
[28] WEI H, YE D, LIU Z, et al. Boosting offline reinforcement learning with residual generative modeling [C]//Thirtieth International Joint Conference on Artificial Intelligence. IJCAI, 2021: 3574 – 3580. DOI: 10.24963/ijcai.2021/492
[29] WANG Z, HUNT J J, ZHOU M. Diffusion policies as an expressive policy class for offline reinforcement learning [EB/OL]. (2022-08-12) [2023-05-15]. https://arxiv.org/abs/2208.06193
[30] CHEN H, LU C, YING C, et al. Offline reinforcement learning via high-fidelity generative behavior modeling [EB/OL]. (2022-09-29) [2023-05-10]. https://arxiv.org/abs/2209.14548
[31] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing atari with deep reinforcement learning [EB/OL]. (2013-12-19)[2022-09-03]. https://arxiv.org/abs/1312.5602
[32] SZITA I, LÖRINCZ A. Learning tetris using the noisy cross-entropy method [J]. Neural computation, 2006, 18(12): 2936 – 2941. DOI: 10.1162/neco.2006.18.12.2936
[33] SOHN K, YAN X, LEE H, et al. Learning structured output representation using deep conditional generative models [C]//28th International Conference on Neural Information Processing Systems. NIPS, 2015: 3483 – 3491
[34] FU J, KUMAR A, NACHUM O, et al. D4RL: datasets for deep data-driven reinforcement learning [EB/OL]. (2020-04-15)[2022-08-15]. https://arxiv.org/abs/2004.07219
[35] HAARNOJA T, ZHOU A, ABBEEL P, et al. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor [C]//International Conference on Machine Learning. Association for Computing Machinery, 2018. DOI: 10.48550/arXiv.1801.01290
[36] FUJIMOTO S, GU S S. A minimalist approach to offline reinforcement learning [EB/OL]. (2021-06-12)[2022-09-13]. https://arxiv.org/abs/2106.06860
[37] TARASOV D, NIKULIN A, AKIMOV D, et al. CORL: research-oriented deep offline reinforcement learning library [C]//3rd Offline Reinforcement Learning Workshop: Offline RL as a "Launchpad". NeurIPS, 2022. DOI: 10.48550/arXiv.2210.07105

Biographies

SHEN Jiahao is currently a postgraduate student in College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. His research interests include reinforcement learning and generative model.

JIANG Ke is currently a PhD student in the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. His research interest is reinforcement learning.

TAN Xiaoyang (x.tan@nuaa.edu.cn) is currently a faculty member of Department of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. His current research interests include machine learning and pattern recognition, computer vision, and reinforcement learning.